NCSR "DEMOKRITOS"
Institute of Nuclear Physics

UNIVERSITY OF ATHENS
Department of Informatics
& Telecommunications

Cluster Installation Guide in CrossGRID

LCFG for Dummies

NCSR "DEMOKRITOS" – INSTITUTE OF NUCLEAR PHYSICS

UOA – DEPT. OF INFORMATICS & TELECOMMUNICATIONS

Cluster Installation in CrossGRID

LCFG for Dummies **(v1.0)**

Vangelis Floros
UOA – Dept. of Informatics & Telecommunications
(floros@di.uoa.gr)

Christos Markou
NCSR "Demokritos"
(cmarkou@inp.demokritos.gr)

Nikos Mastroyiannopoulos
NCSR "Demokritos"
(nmastro@inp.demokritos.gr)

Table of Contents

Chapter
1

Introduction

The purpose of this guide is to provide a systematic set of instructions on the procedure for setting up a Grid cluster. A typical site may consist of one or more "profile servers" and a number of "clients". In a fully operational site, both the servers and the clients can be installed and configured automatically by one of the servers. When setting up a new site, however, at least one of the machines acting as a server must be configured manually.

Who should read this guide

This document is targeted towards Grid Cluster Administrators, who want to set up a cluster from scratch, without prior knowledge of Grid concepts and principles. This guide is the accumulation of months of experience setting-up, upgrading and administering LCFG controlled Grid cluster installations, in the context of "CrossGrid" [1] IST Project.

At the present time, there are at least half a dozen help documents on the net (primary in European DataGrid and CrossGrid project web sites), describing the process. The problem with these documents is that they don't cover the whole process, require a good background regarding Linux and cluster administration, and, what is even worse, they sometimes contain erroneous and outdated material. Thus our effort was twofold; on one hand to provide an integrated view of all this material, filtering out what works and what doesn't, and on the other hand to document our own experience and the problems that emerged during the process.

We must emphasize that the cluster to be set up is not a stand-alone site. This means that there are also several nodes that perform shared services, which are not site-specific, but, rather, shared by a testbed as a whole. This draft is not going to get into the information, which has to do with these shared services.

[1] http://www.crossgrid.org/

The Players

For setting up a minimum Grid cluster one needs at least 5 PCs. Without getting into many details, a typical site consists of the following types of nodes:

Computing Element

This is the main entry to a local site. It hosts the Gatekeeper Grid service, which together with the Worker Node provides computational resources to the user. This node handles the interaction with the rest of the Grid environment accepting jobs, dispatching them for execution and finally returning the output. This provides a uniform interface to the computational resources it manages.
Formally EDG defines a CE to be the ensemble of a Gatekeeper, one or more Worker Nodes and a batch system. In this guide, we use the term CE to declare only the node running the gatekeeper.

Worker Node

These nodes lay behind a Gatekeeper (Computing Element) and are managed via a local batch system. User defined computational tasks are actually performed on these nodes. The end-user software must, therefore, be installed on them. These nodes do not run any EDG[1] daemons, but do have a few APIs[2] for accessing EDG services and information.

Storage Element

These nodes provide uniform access to large storage spaces. The storage element may control large disk arrays, mass storage systems and the like. This element hides the details of the backend storage systems and provides a uniform interface to the user of the grid.

User Interface

This node runs the User Interface software, which allows the end-user to interact with the EDG and CrossGrid testbeds. Users log into this machine to submit jobs to the Grid and to retrieve the output from those jobs.

LCFG server

This is the host where your cluster is orchestrated by using the LCFG[3] suite, together with various other tools and services like NFS[4] and DHCP[5]. The whole installation of the cluster begins on this server. The set up of the software has to be done manually and all the information for configuring each node is retrieved by this server. Moreover, this is the machine that has to be kept "very clean",

1 European Data Grid

2 Application Programming Interface

3 http://www.lcfg.org

4 Network File System

5 Dynamic Host Configuration Protocol

throughout the whole procedure and after completing setting up the site, and that is why some security issues have to be taken into account.

Pre-Setup Logistics

To start a site within the Grid concept, the minimum hardware requirements are:

- 5 PC units (one for each role described above) – preferably with high speed CPU (> 800MHz), large disk space (> 40 Gbytes) and fair amount of main memory (at least 512 MB). Select Fast Ethernet network cards with proven Linux compatibility. Do not install a demanding video adapter (2 Mb memory is fine), at most you will have to use a text console to audit a node's behaviour. Finally, a floppy disk drive is required in every host. Don't spend money on DVD-ROM drivers, CD Recorders and things like that. They are useless for GRID clusters. It's better to install some extra memory instead.

- 2 monitors – one is for the LCFG server and one for testing the installation in each node, although after setting up the site, one needs none.

- 1 keyboard and 1 mouse

Network card MAC Address

Before starting the whole procedure, you have to get (somehow) the MAC[1] addresses of the Ethernet cards. These addresses are going to be needed later on, when installing the DHCP service. Usually, the MAC addresses are written onto the cards or can be found in some feature fields of the BIOS of the system. If this is not the case, do not panic! We will describe later, how to get these MAC addresses.

Network Access

The machines, which are going to be used in the site, have to have network access. A lot of the software that is going to be used will be downloaded from the Web or via FTP. Of course, there must be network connection between the nodes.

Naming

One should decide on the names of the nodes. This is a crucial first step since a lot depend on that. There are three obvious ways of naming the machines. One is to name each node by using its own functionality incorporated in the name, e.g. for worker node *wnode* or *worknode* or *workern* etc. The other convention is to just give every node a number naming e.g. *n001*, *n002* or *cg001*, *cg002* etc. The third option, and the one that we used, is to give a specific name to the LCFG server, e.g. *lcfg* and then name the other nodes of the cluster using an increasing number sequence, e.g. cg001, cg002, etc. This has the advantage of making software changes to all the

1 Media Access Control Address is based on IEEE 802 standards

nodes easier by writing less software commands and with less probability of making mistakes during configuration. In addition, if you want, after the cluster installation, to write shell scripts to automate some processes, you will appreciate the uniform naming schema of your nodes.

IPs and DNS

After dealing with the naming schema of your machines, you must reserve a number of IP addresses for them. Usually, this is something that has to be decided in cooperation with the administrator of the network of your organization. It is important to have in mind the possible growth of the number of the nodes in the cluster in the future. Before start doing anything, it is better to ensure that the names and relative IPs of your hosts have been properly declared in your site's DNS[1] server table information and have been publicly distributed. The *nslookup* program can come in handy for that purpose.

In addition, it is advisable when asking for IP addresses to try to get a contiguous block of addresses, especially if you plan to add more nodes in the future. This will make the firewall configuration easier. If there is shortage of available IPs in a site (something very common nowadays), it is possible to have the WNs without direct network connectivity by utilizing NAT[2].

1 Domain Name System

2 Network Address Translation (NAT) is a technique used to map a set of private IP addresses within an internal network to another set of external IP addresses on a public network.

Chapter

2

LCFG Server Installation

Setting up an LCFG server host can be a tricky endeavour. Since there are many packages to be installed manually or semi-automatically, with much interdependence between them, it is very important to follow the correct sequence of actions. Keep in mind that, most of the time, the installation and administration of one's cluster will be performed through the LCFG server, so one has to make sure that everything works properly.

Installing Red Hat Linux 6.2

Start the Red Hat 6.2 installation by inserting the CD in the drive. In case there is no CD available, one has to get access to the version of Red Hat from a web server. The installation uses a graphical installer named Anaconda.

Hard Disk Partitions

The first decision one has to make is the partitioning of the hard disk. We prefer to keep things simple here and create three Linux native partitions and a Linux swap partition. It is advisable to keep the Linux swap partition as large as the physical memory of the system. The other three partitions are:

/boot, which does not have to be more than 32Mb. 16MB would do fine for most cases. This is the directory where the kernel is stored, and where LILO[1] seeks for it when the system boots up. It must be the first partition in the system.

/opt, which is the partition where all the Grid software will be stored, and from where it will be retrieved later on, for the installation on the cluster nodes. This has to be as large as possible. We suggest reserving 70% of one's total hard disk space for this.

1 Linux Loader

/, which is the root partition and its size is of no real interest, though it is preferable not to be small. We suggest something between 5GB and 30% of your hard disk's space. This is where the local applications are stored, so some future requirements should be taken into consideration.

Of course, you have the freedom to create as many partitions as you like, with variable sizes, but the above configuration is usually sufficient for most of the cases.

Network Configuration

Here you have to fill the configuration of the network IPs, that is:

IP Address:

Netmask:

Network:

Broadcast:

Hostname:

Gateway:

Primary DNS:

Secondary DNS:

The *IP address* is the address of the LCFG server on the network and *hostname* is of course the name you have chosen for your host. To get some of the other information you may need the help of your local site administrator.

Time Zone selection

You must tick the box *system clock uses UTC* and choose your time zone. In the case of NCSR "Demokritos" that is Europe/Athens. The system clock must be in UTC time because that is what LCFG install uses. Grid software (especially Globus[1]) requires all node clocks to be synchronized. By using UTC for the hardware clock and an NTP server[2], as we will see in a later paragraph, one may have machines synchronized, irrespective of their world location. If in a site there is no NTP server configured, you can use one of the publicly available servers around the world[3].

1 http://www.globus.org

2 It is a server which has access to time synchronized by atomic clocks and can distribute this info over the web, via a specific process

3 For more information check: http://www.eecis.udel.edu/~mills/ntp/servers.html

Account configuration

Choose a root password and create another user account, which might be of need in case of an emergency!

Package Group selection

In this section, you have a choice of pre-configured installation profiles. Ignore them and choose *Custom Install*. Then from the given list, select the following packages:

- Printer support
- X window system
- KDE or GNOME[1]
- Mail/WWW/news tools
- DOS/Windows connectivity
- Networked workstations
- NFS server
- Web server
- Network management workstation
- Authoring/publishing
- Emacs
- Development
- Kernel development
- Utilities

Monitor Configuration

Usually Linux gets the most appropriate configuration for your monitor. In any other case, choose according to your monitor settings.

X Configuration

Linux usually finds the graphic card of your system. If it doesn't work, you have to choose the appropriate one by yourself.

Installing Packages

In this step, the system first formats the disk, by (optionally) checking for bad blocks, according to one's opinion and then installs the selected packages. Note

[1] This is not a package that is recommended by DataGrid documentation, but our experience has shown that it is practical to be installed and it doesn't interfere to anything else during the installation

that in a slow machine the check for bad blocks might take quite some time! So, use this option only if you are installing over a used hard disk and you want to be absolutely sure that it is has no defects.

Boot disk creation

It is a good idea to have a Linux boot disk handy, because Murphy is around the corner!

Installing wget

After rebooting, the machine has a clean Linux installation. Before proceeding, you have to install the *wget* package. The *wget* package, which is part of the Red Hat 6.2 release, is a very useful utility that is used to download packages from Web repositories, as the one at IN2P3 Lyon[1]. Since you will be using *wget* a lot, not only during the installation but also throughout the lifetime of your cluster's maintenance and upgrading cycle, it is important to make sure that it is installed properly in your system. The installation can be made either from a web server that has the specific RPM or from the CD ROM version of Linux.

```
$ cd [web server directory or CDROM]/REDHAT/RPMS
$ rpm -iv wget-1.5.3-6.i386.rpm
```

Installing populate-server package

The *populate-server* package distributed by the European DataGrid creates a specific directory tree in the server. The *populate-server* may be downloaded and installed from IN2P3 by the following sequence:

```
$ IN2P3=http://datagrid.in2p3.fr/distribution/datagrid\
/wp4/installation
$ wget $IN2P3/RPMS/LCFG/populate-server-1.0-19.edg.i386.rpm
$ rpm –iv populate-server-1.0-19.edg.i386.rpm
```

Note that, at the time this guide is written, the most recent version of *populate-server* is 1.0-19. You should check and download from IN2P3 the latest version available.

The directory tree, which is created, looks like:

/opt/local/linux/6.2/RPMS/release	RH 6.2 release binary packages
/opt/local/linux/6.2/RPMS/updates	RH 6.2 updates binary packages
/opt/local/linux/6.2/RPMS/external	DataGrid external binary packages
/opt/local/linux/6.2/RPMS/LCFG	DataGrid LCFG binary packages

1 RPM list repository http://datagrid.in2p3.fr/autobuild/rh6.2/rpmlist/

/opt/local/linux/6.2/SRPMS/release	RH 6.2 release source packages
/opt/local/linux/6.2/SRPMS/updates	RH 6.2 updates source packages
/opt/local/linux/6.2/SRPMS/LCFG	DataGrid LCFG source packages
/opt/local/linux/instalroot	LCFG installroot file system

Load Red Hat, OpenSSH and LCFG packages – Extract package headers

In binary and source package directories, the command `make create` loads packages into the directory. In addition, in RH 6.2 directories, the command `make create` copies *release* packages from the CDROM and *update* packages from the web.

Every directory has a Makefile, which must be executed. One can load Red Hat, OpenSSH and LCFG binary packages by:

```
$ cd /opt/local/linux/6.2/RPMS/release
$ make create
$ cd /opt/local/linux/6.2/RPMS/updates
```

Before doing the same thing in the *updates* directory, it is advisable to change the FTP site containing the updates of Red Hat. In the case of NCSR "Demokritos" the site we chose is the FTP site of NTUA. Edit the Makefile changing the SRC variable as follows:

```
SRC=ftp://ftp.ntua.gr/pub/linux/redhat/linux/updates/6.2/en/OS
```

And then from the shell prompt:

```
$ make create
$ cd /opt/local/linux/6.2/RPMS/external
$ make create
$ cd /opt/local/linux/6.2/RPMS/LCFG
$ make create
```

For the installation of LCFG server packages

```
$ cd /opt/local/linux/6.2/RPMS/LCFG
$ make install RPMFLAGS=--force
```

Among other things this `make install` builds and populates two important directories :

/opt/local/linux/6.2/rpmcfg

and

/var/obj/conf/profile/source

These directories contain the configuration files that will be needed for the installation of RPM packages in the nodes and describe the schema of the cluster. We will get back to this later.

During this step, we got warnings for some of the files and at the end, it was suggested to us to execute

```
$ rpm –rebuilddb
```

The most probable cause for these warnings was the existence of multiple versions for some of the programs in the RPMS/LCFG directory. A way around this is to check and move older versions of the programs into a directory `old` and delete the corresponding header files. Care should be taken with the version of the *rpm-package*, which Red Hat uses, since LCFG uses 3.0.5-9 and these two versions have to match. If the version of Red Hat is *rpm-4* or later, it has to be downgraded.

If the previous `make install` has not been successful, there is an impact on the generation of the package header files from the binaries. Due to the above-mentioned warnings, some packages did not install correctly in our case. So when we tried to get the headers:

```
$ cd /opt/local/linux/6.2/RPMS/release
$ make
```

we got the message that the file /usr/sbin/genhdfile does not exist. This utility (`genhdfile`) is the one that gets the header files from the packages and makes .[ ] files. The way around this problem is to install manually the missing RPMs. The `Makefile` in the directory `/opt/local/linux/6.2/RPMS/LCFG` has a list of RPMs under the name `SERVER_RPMS`. Usually the problem is in `rpm*-3.0.5-9.6*.*.rpm` file and every rpm, that is referred from that one and below, is not installed and has to be installed manually:

```
$ rpm –iv [filename]
```

Install updated packages – Kernel update

The procedure of updating the packages is one of the most intriguing and delicate operations that one has to face. For the upgrading, we use the utility *updaterpms*. This program upgrades software that is also used by itself, so extra care should be taken. We have failed twice (!) to make *updaterpms* to work. We had several diagnostic messages and warnings for older versions of software (even though we had moved the older versions in different directories!) and finally we had messages of not being able to find three library files: *Vflib2*, *libVFlib* and *xtt_fonts*. As documented elsewhere[1], one possible solution for this, is to upgrade *glibc* in a separate call to *rpm*, before calling *updaterpms:*

```
$ cd /opt/local/linux/6.2/RPMS/updates
$ rpm -Fv glibc-2.1.3-22.i386.rpm
```

Then you can try to update the whole lot:

```
$ cd /opt/local/linux/6.2/rpmcfg
$ make TARGET=server ROOT=/
```

If everything goes as planned, the kernel version of your Linux will also be upgraded, and these changes should be available to LILO. Therefore, LILO should be instructed to take into account the new version of the kernel.

```
$ /sbin/lilo -v
$ reboot
```

If the upgrade has not completed properly, the kernel must be upgraded manually.

```
$ cd /opt/local/linux/6.2/RPMS/updates
$ rpm -iv
```

Setup installroot, DHCP, NFS and web server

To set up *installroot* the `Makefile` within the corresponding directory must be executed, and then, a soft link of the Red Hat directory to the directory `/ir62` must be established:

```
$ cd /opt/local/linux/installroot
$ make
$ cd /
$ ln -s /opt/local/linux/installroot/6.2 ir62
```

The symbolic link `/ir62` which is created, is used by DHCP entries for backward compatibility reasons with the BOOTP protocol. The DHCP is the Dynamic Host Configuration Protocol and it is used to control vital networking parameters of clients, with the help of a server. For more information, one can consult RFC 1541 and the FAQ[2] on DHCP Server.

The configuration of DHCP starts with the creation of a `/etc/dhcpd.conf` file. Below, is a sample of this file, as it's been used in our LCFG server installation:

1 http://datagrid.in2p3.fr/distribution/datagrid/wp4/installation/doc/server-install-cookbook/linuxinst26.html

2 http://web.syr.edu/~jmwobus/comfaqs/dhcp.faq.html

```
# /etc/dhcpd.conf - DHCP daemon configuration file for dhcpd 2.0
#
# CrossGrid Testbed
# LCFG installation test
# Demokritos, 1/10/2002

deny unknown-clients;
option domain-name "inp.demokritos.gr";

subnet 143.233.250.0 netmask 255.255.255.0 {
option routers 143.233.250.1;
#option domain-name-servers 143.233.7.10;
option domain-name-servers 143.233.250.123;
#option time-servers ntp1.sp.se;

# CE
host xg001 {
        hardware ethernet 00:50:da:82:ca:8d;
        fixed-address 143.233.250.74;
        option root-path "/ir62";
        option option-151 "http://lcfg";
}

# WN
host xg002 {
        hardware ethernet 00:50:da:82:ca:8b;
        fixed-address 143.233.250.75;
        option root-path "/ir62";
        option option-151 "http://lcfg";
}

# SE
host xg003 {
        hardware ethernet 00:50:da:82:ca:8c;
        fixed-address 143.233.250.76;
        option root-path "/ir62";
        option option-151 "http://lcfg";
}

# UI
host xg004 {
        hardware ethernet 00:50:da:82:ce:a4;
        fixed-address 143.233.250.77;
        option root-path "/ir62";
        option option-151 "http://lcfg";
}

} # end subnet
```

Here are a few comments on this file. The hardware Ethernet field is filled with the Ethernet MAC address, which was mentioned in the start of the whole procedure. More info on this, in §2.2. The root path shows the symbolic link of the *installroot* directory and the option-151 defines the web server (LCFG) where the nodes will find their configuration files. Afterwards, a `dhcpd.leases` file must be created.

```
$ touch /var/state/dhcp/dhcpd.leases
```

Then the DHCP daemon has to be started:

```
$ /etc/rc.d/init.d/dhcpd start
```

We should note that other DHCP servers in the network may answer to the DHCP requests of the machines being installed, and this may cause problems. Preferably, the machines should be in a network without other DHCP servers, or the other DHCP servers should be configured not to answer to the requests coming from the machines being managed with LCFG.

The NFS set up is much simpler. We create an export file (`/etc/exports`) that contains the following line:

```
/opt/local/linux                (ro,no_root_squash)
```

(**Important**: Do not leave any spaces between commas and parentheses)

and then we start NFS daemons:

```
$ /etc/rc.d/init.d/nfs start
```

The configuration of the web server starts by stopping the apache web server that is running:

```
$ /etc/rc.d/init.d/httpd stop
```

Then, we edit its configuration file and make its root directory point to LCFG profile and status. Therefore, we replace the strikethrough lines with the normal ones:

~~DocumentRoot "/home/httpd/html"~~

↓

DocumentRoot "/var/obj/conf/profile/web"

~~<Directory "/home/httpd/html">~~

↓

<Directory "/var/obj/conf/profile/web">

Then we restart the web server:

```
$ /etc/rc.d/init.d/httpd start
```

Completing LCFG installation

Define UDP ports

The services of LCFG use some ports that have to be defined. These ports are used by *mkxprof* and *rdxprof* daemons, or in other words, by the notification and acknowledgment protocols. For the definition, we have to add the following lines to `/etc/services`:

```
lcfg              732/udp
lcfgack           733/udp
```

These numbers are not registered which means that we are not constrained to use any other numbers, as long as we are consistent within our site. Beyond this, it is a convention to use the above mentioned ones.

Create LCFG installation disk

The LCFG installation disk can make the installation on any node, existing or to-be, a very easy task. All one needs is a formatted 1.44 MB disk and then write:

```
$ cd /opt/local/linux/6.2/RPMS/LCFG

$ dd if=boot_nfs_28102002.img of=/dev/fd0 bs=5120
```

The `boot_nfs` image file has been downloaded when you run `make create` previously in the LCFG directory. The 28102002 number refers to a date format, for example 28 October 2002.

In addition, this disk is useful of getting the MAC address of the Ethernet cards of the hosts, that we need at the start of the LCFG server installation. If we place the disk in a node – and before continue with the installation – there is a software utility that checks the MAC address of the specific node and prints the information on screen. So, what one can do, in a later addition of a cluster node (or at the beginning of the whole story, if one can borrow such a disk from a previous Grid installation), is to put this disk in the floppy, get the information needed and exit, without doing much else.

`ssh` and security issues

Before completing the LCFG installation, we have to set up `ssh`. Security is one of the main concerns when building large-scale computer clusters. Secure Shell, being one of the most secure ways of communicating between the nodes is an absolute necessity when setting up the cluster. On the LCFG server, `ssh` has to be installed manually. On any other node, the installation is done automatically, since `ssh` is one of the packages, which are scheduled to be installed, as described in the package configuration scripts. `ssh` is an external package, so for the installation we do the following:

```
$ cd /opt/local/linux/6.2/RPMS/edg-1.2.2/external
$ rpm –iv openssl-0.9.5a-24.i386.rpm
$ rpm –iv openssl-devel-0.9.5a-24.i386.rpm
$ rpm –iv openssh-3.4p1-1_edg2.i386.rpm
$ rpm –iv openssh-server_3.4p1-1_edg2.i386.rpm
$ rpm –iv openssh-clients-3.4p1-1_edg2.i386.rpm
```

Chapter 3

Cluster definition in LCFG

LCFG is designed to handle automated installation and configuration in a very diverse and rapidly changing environment. The LCFG server will keep the whole information for the cluster description, the profiles of the client computers and the control of the version synchronization of the used software, between the nodes of the local cluster and that of the remote machines. Figure 1 shows the overall architecture of the LCFG framework:

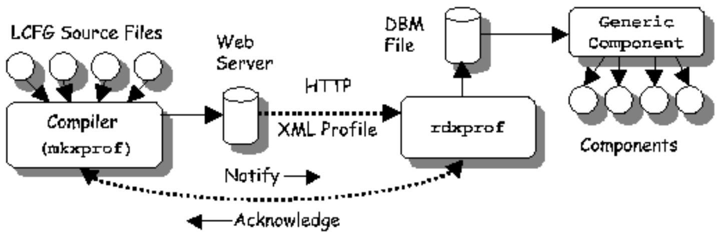


Figure 1 - LCFG architecture

The specific definition and description of the nodes is a job that will free the hands of the future to-be administrator of the cluster. Every change that has to be done, will take place only once and in a specific way, with absolute rules and procedures, which is something that has an overall impact on the broad usability of the specific technology used. The correct definition of the nodes is the ABC of the functionality of your site's cluster.

RPM Repository and Download

As our main repository, we have used the central repository[1] of the DataGrid project. From that place, one can find the lists of all the software that has to be downloaded on to one's site. The current version of `edg` at the time this manual was written is **edg-1_2_2**

[1] http://datagrid.in2p3.fr/autobuild/rh6.2/rpmlist/

Before downloading the specific files, the specific directory tree that will host all the files needed, must be created in the LCFG server. Neither DataGrid nor CrossGrid have any specific directives on the creation of this directory tree. We have opted for a very conservative solution which not only allows us to identify easily the version of `edg` currently running on the server, but also to keep the whole information of the version developing through each stage. This requires a lot of space and it might be a drawback in the future, but it is not necessary to keep all the `edg` software in that way. We have the suggestion of keeping the three latest versions that way, even in directories that the package software doesn't change so often (e.g. rh6.2) and delete older versions (directories). Therefore, we create a directory with the name of the edg currently installing

```
$ cd /opt/local/linux/6.2/RPMS
$ mkdir edg-1_2_2
```

Then into this directory, we create all the directories that we will need as they are documented in the above-mentioned web link. These directories are:

```
$ mkdir ca
$ mkdir edg
$ mkdir globus
$ mkdir external
$ mkdir rh6.2
$ mkdir lcfg
$ mkdir appcommon
$ mkdir appwp8
$ mkdir appwp9
$ mkdir appwp10
$ mkdir invalid
```

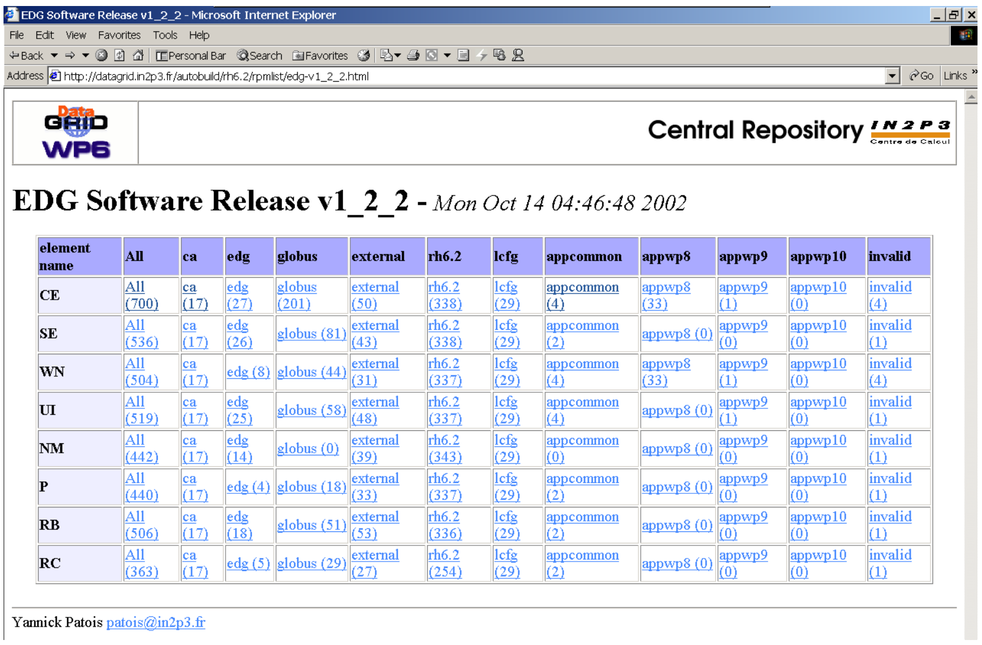
Any new version of `edg` will be installed in the same way in the appropriate directory, e.g. `edg_1_2_3` with the same directory tree under it.

The next step is to copy the `Makefile` into the directories that have just being created. This can be found in the directory /opt/local/linux/6.2/RPMS/release.

Then from the command prompt and using the *wget* utility, we have to download the RPMs to populate the repositories on the LCFG server. Within each directory, we execute the following command

```
$ wget -r -nd <URL>
```

The URL is the one referring to the specific web address where the rpm list exists e.g. for the 1-2-2 version of `edg`, the `ca` of the Computing Element is: http://datagrid.in2p3.fr/autobuild/rh6.2/rpmlist/CE-ca-v1_2_2.html. The RPM packages are not the same for every element of the cluster, thus the packages referring to every separate element must be downloaded. In the scheme we have followed, the directory structure does not depend on the specific element. Consequently, the packages for each element will be downloaded in the same directories mentioned above. A snapshot of the site of the European DataGrid software release v1_2_2 appears on Figure 2.


EDG Software Release v1_2_2 - Microsoft Internet Explorer

http://datagrid.in2p3.fr/autobuild/rh6.2/rpmlist/edg-v1_2_2.html

Central Repository

EDG Software Release v1_2_2 - *Mon Oct 14 04:46:48 2002*

element name	All	ca	edg	globus	external	rh6.2	lcfg	appcommon	appwp8	appwp9	appwp10	invalid
CE	All (700)	ca (17)	edg (27)	globus (201)	external (50)	rh6.2 (338)	lcfg (29)	appcommon (4)	appwp8 (33)	appwp9 (1)	appwp10 (0)	invalid (4)
SE	All (536)	ca (17)	edg (26)	globus (81)	external (43)	rh6.2 (338)	lcfg (29)	appcommon (2)	appwp8 (0)	appwp9 (0)	appwp10 (0)	invalid (1)
WN	All (504)	ca (17)	edg (8)	globus (44)	external (31)	rh6.2 (337)	lcfg (29)	appcommon (4)	appwp8 (33)	appwp9 (1)	appwp10 (0)	invalid (4)
UI	All (519)	ca (17)	edg (25)	globus (58)	external (48)	rh6.2 (337)	lcfg (29)	appcommon (4)	appwp8 (0)	appwp9 (1)	appwp10 (0)	invalid (1)
NM	All (442)	ca (17)	edg (14)	globus (0)	external (39)	rh6.2 (343)	lcfg (29)	appcommon (0)	appwp8 (0)	appwp9 (0)	appwp10 (0)	invalid (1)
P	All (440)	ca (17)	edg (4)	globus (18)	external (33)	rh6.2 (337)	lcfg (29)	appcommon (2)	appwp8 (0)	appwp9 (0)	appwp10 (0)	invalid (1)
RB	All (506)	ca (17)	edg (18)	globus (51)	external (53)	rh6.2 (336)	lcfg (29)	appcommon (2)	appwp8 (0)	appwp9 (0)	appwp10 (0)	invalid (1)
RC	All (363)	ca (17)	edg (5)	globus (29)	external (27)	rh6.2 (254)	lcfg (29)	appcommon (2)	appwp8 (0)	appwp9 (0)	appwp10 (0)	invalid (1)

Yannick Patois patois@in2p3.fr

Figure 2 - DataGRID Central Repository

So, the correct sequence is to download the packages of the first element (CE), then those of the second one (SE), and so on. In order to download the packages that have not yet been downloaded, the following command must be issued:

```
$ wget –r –nd –nc <URL>
```

The packages have to be downloaded for the elements: CE, SE, WN and UI.

Then after finishing the population of the repositories for each directory, the HDFILES must be updated.

```
$ cd /opt/local/linux/RPMS/<directory>
$ make
```

It should be noted that each time new RPMs are added in the repositories, the command `make` must be issued in the repositories that have changed. This is necessary in order to create afresh the header files which are needed by LCFG.

Finally, some country specific RPMs must be included in the ca directory. These must be inserted manually and are related with the certificates of the country. These RPMs can be obtained from the organisation, which is responsible for issuing the authentication certificates of the country.

RPM List Customisation

There are two kinds of RPM lists. These exist in two different directories of the LCFG server.

Retrieving RPM Lists

The main source of retrieving the RPM lists is the CVS repository of CrossGRID[1]. From there, you can follow the links to cg-interim-inst → crossgrid → sites → lip or fzk. There are two kinds of files. Source and rpmcfg files. The source files are downloaded with CVS commands to the local directory /var/obj/conf/profile/source. Only the files referring to your specific EDG version should be downloaded. The other ones are downloaded to the directory of LCFG /opt/local/linux/6.2/rpmcfg.

After retrieving these files the directory /opt/local/linux/6.2/rpmcfg looks like this:

```
/opt/local/linux/6.2/rpmcfg
total 168
-rw-r--r--    1 root     root          122 Sep 17 13:10 Alice-rpm.h
-rw-r--r--    1 root     root          632 Sep 17 13:10 Atlas-rpm.h
-rw-r--r--    1 root     root         2916 Oct  3 11:08 CE-rpm
-rw-r--r--    1 root     root          184 Sep 17 13:10 CMS-rpm.h
-rw-r--r--    1 root     root          391 Sep 17 13:10 ComputingElement-OPTIONAL-rpm.h
-rw-r--r--    1 root     root         9735 Sep 17 13:10 ComputingElement-rpm.h
-rw-r--r--    1 root     root          154 Sep 17 13:10 DZero-rpm.h
-rw-r--r--    1 root     root          121 Sep 17 13:10 LHCb-rpm.h
-rw-r--r--    1 root     root         2390 Sep 17 13:10 NM-rpm
-rw-r--r--    1 root     root          667 Sep 17 13:10 NetworkMonitoring-rpm.h
-rw-r--r--    1 root     root         2326 Sep 17 13:10 Other-rpm
-rw-r--r--    1 root     root         2204 Sep 17 13:10 P-rpm
-rw-r--r--    1 root     root         1227 Sep 17 13:10 Proxy-rpm.h
-rw-r--r--    1 root     root         2362 Sep 17 13:10 RB-rpm
-rw-r--r--    1 root     root         2325 Sep 17 13:10 RC-rpm
-rw-r--r--    1 root     root         1685 Sep 17 13:10 ReplicaCatalog-rpm.h
-rw-r--r--    1 root     root         3698 Sep 17 13:10 ResourceBroker-rpm.h
-rw-r--r--    1 root     root         2535 Oct  3 11:33 SE-rpm
-rw-r--r--    1 root     root          369 Sep 17 13:10 StorageElement-OPTIONAL-rpm.h
-rw-r--r--    1 root     root         4806 Sep 17 13:10 StorageElement-rpm.h
-rw-r--r--    1 root     root         2277 Sep 17 13:10 UI-rpm
-rw-r--r--    1 root     root         3603 Sep 17 13:10 UserInterface-rpm.h
-rw-r--r--    1 root     root         2897 Oct  3 12:03 WN-rpm
-rw-r--r--    1 root     root           95 Sep 17 13:10 WP9-rpm.h
-rw-r--r--    1 root     root          348 Sep 17 13:10 WorkerNode-OPTIONAL-rpm.h
-rw-r--r--    1 root     root         2205 Sep 17 13:10 WorkerNode-rpm.h
-rw-r--r--    1 root     root          205 Sep 17 13:10 apps_common-rpm.h
-rw-r--r--    1 root     root           28 Oct 15 16:18 file1
-rw-r--r--    1 root     root          635 Sep 17 13:10 lcfg-client-rpm.h
-rw-r--r--    1 root     root          609 Sep 17 13:10 lcfg-common-rpm.h
-rw-r--r--    1 root     root         1140 Sep 17 13:10 lcfg-server-rpm.h
-rw-r--r--    1 root     root          165 Sep 17 13:10 openssh-rpm.h
-rw-r--r--    1 root     root        19416 Sep 17 13:10 rh62-release-rpm.h
-rw-r--r--    1 root     root         2415 Sep 17 13:10 rh62-updates-rpm.h
-rw-r--r--    1 root     root          819 Sep 17 13:11 security-rpm.h
```

The source directory looks like this:

```
/var/obj/conf/profile/source
total 424
-rw-r--r--    1 root     root        13504 Aug 30 14:11 CGUsers-cfg.h
-rw-r--r--    1 root     root         1766 Aug 30 15:04 CGUsersNoHome-cfg.h
-rw-r--r--    1 root     root        13838 Sep 13 17:37 ComputingElement-cfg.h
-rw-r--r--    1 root     root          880 Aug 28 13:50 NetworkMonitor-cfg.h
-rw-r--r--    1 root     root         9184 Aug 28 13:50 Proxy-cfg.h
-rw-r--r--    1 root     root        11585 Aug 28 13:50 ResourceBroker-cfg.h
-rw-r--r--    1 root     root        37155 Sep 13 17:34 StorageElement-cfg.h
-rw-r--r--    1 root     root         2191 Aug 30 14:25 UserInterface-cfg.h
-rw-r--r--    1 root     root        97901 Aug 28 13:50 Users-cfg.h
-rw-r--r--    1 root     root          739 Oct  1 13:05 Users-demo-cfg.h
-rw-r--r--    1 root     root        15125 Aug 28 13:50 UsersNoHome-cfg.h
-rw-r--r--    1 root     root          967 Aug 28 14:36 UsersNoHome-uoa-cfg.h
-rw-r--r--    1 root     root         2178 Aug 30 14:27 WorkerNode-cfg.h
```

[1] http://gridportal.fzk.de/cgi-bin/viewcvs.cgi

```
-rw-r--r--    1 root     root        32759 Aug 28 14:51 afs.def
-rw-r--r--    1 root     root         1237 Aug 28 14:51 auth.def
-rw-r--r--    1 root     root         1750 Aug 28 14:51 boot.def
-rw-r--r--    1 root     root         4371 Aug 28 14:51 ccm.def
-rw-r--r--    1 root     root         7145 Oct  1 12:18 client_testbed-cfg.h
-rw-r--r--    1 root     root          670 Aug 28 14:51 cron.def
-rw-r--r--    1 root     root           29 Oct 15 16:23 file2
-rw-r--r--    1 root     root           92 Aug 28 14:51 filecopy.def
-rw-r--r--    1 root     root          517 Aug 28 14:51 gdmp.def
-rw-r--r--    1 root     root           69 Aug 28 14:51 gdmp_client.def
-rw-r--r--    1 root     root           18 Aug 28 14:51 generic.def
-rw-r--r--    1 root     root          895 Aug 28 14:51 globus.def
-rw-r--r--    1 root     root          441 Aug 28 14:51 inet.def
-rw-r--r--    1 root     root          776 Aug 28 14:51 lcas.def
-rw-r--r--    1 root     root          135 Aug 28 14:51 ldconf.def
-rw-r--r--    1 root     root         7375 Oct  3 11:41 linuxdef-cfg.h
-rw-r--r--    1 root     root          483 Aug 28 13:50 macros-cfg.h
-rw-r--r--    1 root     root          279 Aug 28 14:51 mailng.def
-rw-r--r--    1 root     root          361 Aug 28 14:51 nfs.def
-rw-r--r--    1 root     root          266 Aug 28 14:51 nfsmount.def
-rw-r--r--    1 root     root          646 Aug 28 14:51 nsswitch.def
-rw-r--r--    1 root     root          561 Aug 28 14:51 pam.def
-rw-r--r--    1 root     root         1057 Aug 28 13:50 pbs-cfg.h
-rw-r--r--    1 root     root         2055 Aug 28 14:51 profile.def
-rw-r--r--    1 root     root          283 Aug 28 14:51 rms.def
-rw-r--r--    1 root     root        12761 Oct  3 10:53 site-cfg.h
-rw-r--r--    1 root     root          402 Aug 28 14:51 syslog.def
-rw-r--r--    1 root     root          999 Aug 28 14:51 update.def
-rw-r--r--    1 root     root          808 Aug 28 14:51 updaterpms.def
-rw-r--r--    1 root     root          692 Oct  3 15:57 xg001
-rw-r--r--    1 root     root          825 Oct  3 15:55 xg002
-rw-r--r--    1 root     root          785 Oct  3 15:58 xg003
-rw-r--r--    1 root     root          583 Oct  3 15:59 xg004
-rw-r--r--    1 root     root         1050 Aug 28 14:51 xntpd.def
```

This directory has two kinds of files. The `*.def` files, which are default configuration files required for each component to list its resources together with any default values and `*.h` files, which are header files used to hold groups of resources which are common to sets of machines.

Configuration of RPM Lists

The files of rpmcfg include the rpm package lists of the repository directories. Each list corresponds to a specific type of machine. The proposed layout is a file with a high-level list that includes basic lists. Some of these basic lists can be shared by different type of nodes. For example, the high level list for the Computing Element is `CE-rpm`. The `CE-rpm` has the following structure:

```
/*
  CE-rpm
  ===============================================
  ComputingElement high-level RPM list
*/

/* Select Red Hat 6.2 release (Anaconda) package groups */

#define RH_Base 1
#define RH_Printer_Support 1
#define RH_X_Window_System 1
#define RH_GNOME 0
#define RH_KDE 0
#define RH_Mail_WWW_News_Tools 1
#define RH_DOS_Windows_Connectivity 1
#define RH_Graphics_Manipulation 0
#define RH_Games 0
#define RH_Multimedia_Support 0
#define RH_Networked_Workstation 1
#define RH_Dialup_Workstation 0
#define RH_News_Server 0
#define RH_NFS_Server 0
#define RH_SMB_Samba_Server 0
#define RH_IPX_Netware_Connectivity 0
#define RH_Anonymous_FTP_Server 0
#define RH_Web_Server 0
```

```
#define RH_DNS_Name_Server 0
#define RH_Postgres_SQL_Server 0
#define RH_Network_Management_Workstation 0
#define RH_Authoring_Publishing 1
#define RH_Emacs 1
#define RH_Development 1
#define RH_Kernel_Development 1
#define RH_Clustering 0
#define RH_Utilities 1

/* Hidden package groups */
#define RH_Network_Server 1
#define RH_Workstation_Common 0
#define RH_GNOME_Workstation 0
#define RH_KDE_Workstation 0
#define RH_Server 0

/* Define architecture */
#define ISARCH_i386 1
#define ARCH i686

/* RedHat 6.2 + updates */
#include "rh62-release-rpm.h"
#include "rh62-updates-rpm.h"

/* LCFG support */
#include "lcfg-common-rpm.h"
#include "lcfg-client-rpm.h"

/* OpenSSH */
#include "openssh-rpm.h"

/* CA certificates */
#include "security-rpm.h"

#define OPENPBS 1
/* ComputingElement specific packages */
#include "ComputingElement-rpm.h"

/* This file contains software which is OPTIONAL.  It is here to
   facilitate interoperability tests with other projects, but is in no
   way required to get the EDG software up and running.  You can
   add these by uncommenting the line below. */
/* #include "ComputingElement-OPTIONAL-rpm.h" */

/* Common experiment software */
#include "apps_common-rpm.h"

/* CMS software */
#include "CMS-rpm.h"

/* Atlas software */
/* #include "Atlas-rpm.h" */

/* Alice software */
/* #include "Alice-rpm.h" */

/* LHCb software */
/* #include "LHCb-rpm.h" */

/* WP9 software */
/* #include "WP9-rpm.h" */

/* DZero software */
/* #include "DZero-rpm.h" */

/* Packages added to satisfy dependencies */
libxml-1.8.6-2
/* Red Hat 6.2 update packages which require rpm 4 */
/* +rhn_register-*-*:i */
+up2date-*-*:i
/* Packages whose trigger scripts go wrong during installation */
+linuxconf-1.17r2-6:i
+linuxconf-devel-1.17r2-6:i
/* Packages whose postinstall scripts go wrong during installation */
#if RH_Web_Server
  +mod_perl-1.23-3:s
  +php-3.0.18-1:s
#endif
+jade-1.2.1-9:s
+sgml-common-0.1-7:s
+docbook-3.1-3:s
+stylesheets-0.13rh-4:s
```

```
+openldap-servers-1.2.12-3:s

/* Explicit package versions, to avoid confusion with Edinburgh packages */
+filesystem-1.3.5-1
+setup-2.1.8-1

/* Additional packages not yet selected */
python-devel-1.5.2-13:s
wget-1.5.3-6
XFree86-SVGA-3.3.6-29
```

In this kind of files, there are constructs of the C language (#define, #include etc) that are used for the correct set up of the specific node. In the above example, after the first definition for the standard Linux package installation, there exist the RPMs for the Linux and its updates, the LCFG support, OpenSSH, CA certificates and then the specific package for the node (in this case it is the CE). Then, in the end, there is the software of the VOs[1] that are going to be used. In the above example, the only VO used is for the CMS Experiment so the software of CMS is included. The VO software is not needed in the SE node of the cluster. In addition, the file corresponding to CA certificates has to be changed. As we said before, in the RPMS repository, we put the country specific certificates. These certificates have to be registered to the file `security-rpm.h`.

```
/*
  security-rpm.h
  ================================================
  DATAGRID CA CERTIFICATES
*/

/* !! WARNING !!
   Uncomment the correct package (only 1 package!) for your organization
   from these ca_*-local RPMs */

/* ca_CERN-local-0.10-1 */
/* ca_GermanGrid-local-0.10-1 */
/* ca_GridPP-local-0.10-1 */
/* ca_INFN-local-0.10-1 */
/* ca_NorduGrid-local-0.10-1 */
/* ca_Russia-local-0.10-1 */

ca_CERN-0.10-1
ca_CERN-new-0.10-1
ca_CESNET-0.10-1
ca_CNRS-0.10-1
ca_CNRS-DataGrid-0.10-1
ca_CNRS-Projets-0.10-1
ca_DOESG-0.10-1
ca_DOESG-Root-0.10-1
ca_GermanGrid-0.10-1
ca_Grid-Ireland-0.10-1
ca_GridPP-0.10-1
ca_INFN-0.10-1
ca_LIP-0.10-1
ca_NIKHEF-0.10-1
ca_NorduGrid-0.10-1
ca_Russia-0.10-1
ca_Spain-0.10-1

/* Two additional rpms added for HellasGrid CA
*/
ca_HellasGrid-0.1-1
ca_HellasGrid-local-0.1-1
```

In our case, we added the country certificates at the end of the file.

[1] VOs or Virtual Organizations are logical views of the Grid communities organized by area of activity.

Profile Configuration

The configuration of the profile is going to be broadcasted from the LCFG server – playing the role of a web server – through httpd to the hosts. There is one file for each host. The name of the file must be the hostname of the host. Normally one should only set the machine's name and some other configuration files. In our site, the names of the hosts and their corresponding files are: xg001, xg002, xg003, xg004 corresponding to CE, WN, SE and UI. The structure of our xg001 is the following:

```
/*
xg001 - LCFG profile for CrossGrid
(an LCFG client computer - Computing Element)
October 02,2002
*/

/* Host specific definitions */
#define HOSTNAME xg001

/* Some useful macros */
#include "macros-cfg.h"

/* Site specific definitions */
#include "site-cfg.h"

/* Linux default resources */
#include "linuxdef-cfg.h"

/* LCFG client specific resources */
#include "client_testbed-cfg.h"

/* Users required by VOs (Atlas, CMS etc) */
#include "Users-cfg.h"

/* Users for CrossGrid */
#include "CGUsers-cfg.h"

/* Computing Element specific resources */
#include "ComputingElement-cfg.h"

/* PBS */
#include "pbs-cfg.h"
```

We define the name of the host followed by some header files. These header files need to be adapted for each specific site.

The `macros-cfg.h` file is a file with some common macros that are used by all the hosts. The `site-cfg.h` header file, is a site-specific file and it is one of the most important files in the source directory. The NCSR "Demokritos" `site-cfg.h` file appears below:

```
/*
  site-cfg.h.UoA/Demokritos
  ==================================================
  SITE SPECIFIC CONFIGURATION
*/

/* COMMON LCFG DEFINITIONS -----------------------------------------------
--------------------------------------------------------------------------- */
#define LCFGSRV                     lcfg
#define URL_SERVER_CONFIG           http://lcfg

/* COMMON GRID DEFINITIONS -----------------------------------------------
--------------------------------------------------------------------------- */
```

```
/* The host and port number of the information index associated with
   the resource broker.  These are testbed013.cern.ch and 2170 for the
   development testbed.  Change these to appropriate values.  This is
   used by the gdmp object to fetch the list of SE subject names
   reachable from the broker. */
#define GRID_RB_II_HOST         lngrid06.lip.pt
#define GRID_RB_II_PORT         2135

/* COMMON SITE DEFINITIONS -----------------------------------------------
------------------------------------------------------------------------- */
#define LOCALDOMAIN             inp.demokritos.gr
#define SITE_MAILROOT           nmastro@inp.demokritos.gr
#define SITE_GATEWAYS           143.233.250.1
/* Allowed networks (useful for tcpwrappers) */
#define SITE_ALLOWED_NETWORKS   127.0.0.1, 143.233.250.
#define SITE_NAMESERVERS        143.233.250.123 143.233.7.10
/* The netmask */
#define SITE_NETMASK            255.255.255.0
/* NTP server (domain and hostname) */
#define SITE_NTP_DOMAIN         sp.se
#define SITE_NTP_HOSTNAME       ntp1
/* The time zone */
#define SITE_TIMEZONE           Europe/Athens
/* Site name */
#define SITE_NAME_              Demokritos
/* Site EDG version */
#define SITE_EDG_VERSION        edg1_2_2
/* Site installation date */
#define SITE_INSTALLATION_DATE_ 20021001111300Z
/* Site distinguished name. */
#define SITE_DN_                \"dc=inp, dc=demokritos, dc=gr, o=DemoGrid\"
/* All the WN (used by /etc/export configuration of /home NFS Mount
   e.g. testbed*.lnl.infn.it. Needed by ComputingElement.h) */
#define SITE_WN_HOSTS           xg002.inp.demokritos.gr
/* All the SE hosts (comma separated list) */
#define SITE_SE_HOSTS_          SE_HOSTNAME
/* List (comma separated) of the Computing Element(s) of your site */
#define SITE_CE_HOSTS_          CE_HOSTNAME:2119/jobmanager-pbs-workq
/* The following section covers the setup of the GIIS hierarchy
   The first two lines specify the information for where the machine-local
   GRISes should register; this is the SITE GIIS (e.g. "nikhef" or "ral")
   first the "VO name" (name of GIIS) */
#define SITE_GIIS               demokritos
/* next the hostname where this IIS runs */
#define SITE_GIIS_HOSTNAME      xg001.inp.demokritos.gr

/* There are two supported options: SITE GIIS on an external machine
   (meaning not under LCFG control), or SITE GIIS on the CE.  In the
   second case, uncomment the next three lines which handle starting
   the SITE GIIS and specifying to which GIIS it should register
   turns on or off SITE GIIS on CE machine */
#define SITE_GIIS_ON_CE
/* what is the name of the GIIS (country or organization GIIS) to which
   SITE GIIS should register? */

#define COUNTRY_OR_ORG_GIIS       xg001.inp.demokritos.gr
/* where is that GIIS running? */
#define COUNTRY_OR_ORG_GIIS_HOSTNAME  xg001.inp.demokritos.gr

/* Some sites may choose to run the country or org GIIS on the CE as well
   if so, comment out the following three lines meaning similar to the
   last three turns on or off COUNTRY/ORG GIIS on CE machine */
/* #define COUNTRY_GIIS_ON_CE */
/* register COUNTRY GIIS to datagrid GIIS */
/* SITE GIIS should register? */
#define TOP_GIIS           datagrid
/* where is that GIIS running? */
#define TOP_GIIS_HOSTNAME  lxshare0225.cern.ch

/* COMPUTING ELEMENT DEFINITIONS -----------------------------------------
------------------------------------------------------------------------- */
/* ComputingElement hostname */
#define CE_HOSTNAME             xg001.inp.demokritos.gr
/* Subject of the certificate */
#define CE_CERT_SBJ
\"/C=GR/O=HellasGrid/OU=inp.demokritos.gr/CN=xg001.inp.demokritos.gr\"
/* Some site and host information (it goes in globus.conf)*/
#define CE_HOST_DN              \"hn=xg001.inp.demokritos.gr, dc=inp, dc=demokritos,
dc=gr, o=DemoGrid\"
/* Full path of the certificate */
#define CE_CERT_PATH            /opt/globus/etc/xg001-computingelement.cert
/* Full path of the secret key */
#define CE_SECKEY_PATH          /opt/globus/etc/xg001-computingelement.key
```

```
/* System administrator e-mail */
#define CE_SYSADMIN             nmastro@inp.demokritos.gr
/* Space separed job managers list (e.g. fork, pbs, lsf) (this info goes in
globus.conf) */
#define CE_JOBMANAGERS          \"fork pbs\"
/* Batch system adopted by CE (this info goes in info-mds.conf */
#define CE_BATCHSYSTEM_         pbs
/* Binaries path of the batch system */
#define CE_BATCHSYSTEM_BIN_PATH /usr/pbs/bin
/* Local queue names */
#define CE_QUEUE_               workq
/* List (comma separated without spaces) of StorageElement(s) close to this CE */
#define CE_CLOSE_SE_ID_         SE_HOSTNAME
/* Mount point(s) of the SE(s) close to this CE */
#define CE_CLOSE_SE_MOUNTPOINT  /flatfiles
/* Disk description */
#define CE_DISK_DESC            18GB-EIDE
/* CPU description */
#define CE_CPU_DESC             P3-550MHz
/* CE InformationProviders: MinPhysMemory */
#define CE_IP_MINPHYSMEM        512
/* CE InformationProviders: MinLocalDiskSpace */
#define CE_IP_MINLOCDISK        2048
/* CE InformationProviders: NumSMPs */
#define CE_IP_NUMSMPS           1
/* CE InformationProviders: MinSPUProcessors */
#define CE_IP_MINSPUPROC        1
/* CE InformationProviders: MaxSPUProcessors */
#define CE_IP_MAXSPUPROC        1
/* CE InformationProviders: MaxSI00. See some examples of SpecInt at
   http://www.specbench.org/osg/cpu2000/results/cint2000.html */
#define CE_IP_MAXSI00           390
/* CE InformationProviders: MinSI00 */
#define CE_IP_MINSI00           390
/* CE InformationProviders: AverageSI00 */
#define CE_IP_AVRSI00           390
/* CE InformationProviders: AFSAvailable: */
#define CE_IP_AFS_AFSAVAILABLE  FALSE
/* CE InformationProviders: OutboundIP */
#define CE_IP_OUTBOUNDIP        TRUE
/* CE InformationProviders: InboundIP */
#define CE_IP_INBOUNDIP         TRUE
/* CE InformationProviders: RunTimeEnvironment (1) */
#define CE_IP_RUNTIMEENV1       CMS-1.1.0
/* CE InformationProviders: RunTimeEnvironment (2) */
#define CE_IP_RUNTIMEENV2       ATLAS-3.2.1
/* CE InformationProviders: RunTimeEnvironment (3) */
#define CE_IP_RUNTIMEENV3       ALICE-3.07.01
/* CE InformationProviders: RunTimeEnvironment (4) */
#define CE_IP_RUNTIMEENV4       LHCb-1.1.1
/* CE InformationProviders: RunTimeEnvironment (5) */
#define CE_IP_RUNTIMEENV5       INFN
/* CE InformationProviders: RunTimeEnvironment (6) */
#define CE_IP_RUNTIMEENV6       LNL
/* CE InformationProviders: RunTimeEnvironment (7) */
#define CE_IP_RUNTIMEENV7       CMSIM-125
/* CE InformationProviders: RunTimeEnvironment (8) */
/*#define CE_IP_RUNTIMEENV8       ORCA-6.0.2 */
/* CE InformationProviders: RunTimeEnvironment (9) */
/*#define CE_IP_RUNTIMEENV9       !uncomment and define it if you need! */
/* CE InformationProviders: RunTimeEnvironment (10) */
/*#define CE_IP_RUNTIMEENV10      !uncomment and define it if you need! */
/* CE InformationProviders: RunTimeEnvironment (11) */
/*#define CE_IP_RUNTIMEENV11      !uncomment and define it if you need! */
/* CE InformationProviders: RunTimeEnvironment (12) */
/*#define CE_IP_RUNTIMEENV12      !uncomment and define it if you need! */
/* CE InformationProviders: RunTimeEnvironment (13) */
/*#define CE_IP_RUNTIMEENV13      !uncomment and define it if you need! */
/* CE InformationProviders: RunTimeEnvironment (14) */
/*#define CE_IP_RUNTIMEENV14      !uncomment and define it if you need! */
/* CE InformationProviders: RunTimeEnvironment (15)
   This must be defined for your CE; it indicates that your site is running
   but hasn't yet been certified.  Change this to EDG-CERTIFIED once your
   site has been tested by the ITeam. */
#define CE_IP_RUNTIMEENV15      EDG-TEST
/*#define CE_IP_RUNTIMEENV15      EDG-CERTIFIED */

/* The mountpoint on the CE of the SE exported area via NFS */
#define CE_MOUNTPOINT_SE_AREA   /flatfiles

/* Uncomment this below if you want to collect and publish
   data from a network monitor */
/* #define NETMON_HOST_ gppnm06.gridpp.rl.ac.uk */
/* STORAGE ELEMENT DEFINITIONS ------------------------------------------
---------------------------------------------------------------------------- */
/* StorageElement hostname */
#define SE_HOSTNAME             xg003.inp.demokritos.gr
/* Full path of the certificate */
#define SE_CERT_PATH            /opt/globus/etc/xg003-storageelement.cert
/* Full path of the secret key */
#define SE_SECKEY_PATH          /opt/globus/etc/xg003-storageelement.key
/* Subject of the SE certificate */
#define SE_CERT_SBJ
\"/C=GR/O=HellasGrid/OU=inp.demokritos.gr/CN=xg003.inp.demokritos.gr\"
/* Some site and host information (it goes in globus.conf) */
#define SE_HOST_DN              \"hn=xg003.inp.demokritos.gr, dc=inp, dc=demokritos,
dc=gr, o=DemoGrid\"
/* System administrator e-mail */
#define SE_SYSADMIN             nmastro@inp.demokritos.gr
/* List (comma separated without spaces) of ComputingElement(s) close to the SE. */
#define SE_CLOSE_CE_            CE_HOSTNAME
/* The value of SE_SIZE in info-mds.conf */
#define SE_DISKSIZE             18
/* comma separated list without spaces, values used in df to obtain freespace */
#define SE_FILESYSTEMS_         /dev/hda3
/* Disk description */
#define SE_DISK_DESC            18GB-EIDE
/* CPU description */
#define SE_CPU_DESC             PIII-550MHz
/* SE protocols */
#define SE_PROTOCOLS_           gridftp,rfio,file
/* SE protocols ports */
#define SE_PROTOCOL_PORTS_      2811,3147,
/* GDMP area */
#define SE_GDMP_AREA            /flatfiles
/* List of the supported VO. Add/remove the VO name for each VO that you
   support/do not support */
#define SE_VO_
alice:/gdmp/vo/path,atlas:/gdmp/vo/path,cms:/gdmp/vo/path,lhcb:/gdmp/vo/path,biome:/gdm
p/vo/path,eo:/gdmp/vo/path,wpsix:/gdmp/vo/path,cg:/gdmp/vo/path
/* For each VO that you support, uncomment the corresponding line (you MUST
   uncomment at least 1 line). For each VO that you select, a) 50 accounts
   will be created b) a GDMP server will be configured  */
/*#define SE_VO_ALICE*/
#define SE_VO_ATLAS
#define SE_VO_CMS
/*#define SE_VO_LHCB
#define SE_VO_BIOM
#define SE_VO_EOBS */
#define SE_VO_WPSIX
#define SE_VO_CG
/******/
#define SITE_CREATE_ACCOUNTS_CG
#define SE_GDMP_REP_CAT_CG_PWD Testbed1RC
/* Uncomment this if you want to create local accounts mapped to the Integration Team
*/
/* #define SITE_CREATE_ACCOUNTS_ITEAM */
/* Password for Globus Replica Catalog Manager of Alice. This is COMPULSORY ONLY
   if your SE supports Alice */
#define SE_GDMP_REP_CAT_ALICE_PWD !Top!Secret!
/* Password for Globus Replica Catalog Manager of Atlas. This is COMPULSORY ONLY
   if your SE supports Atlas */
#define SE_GDMP_REP_CAT_ATLAS_PWD !Top!Secret!
/* Password for Globus Replica Catalog Manager of CMS. This is COMPULSORY ONLY
   ONLY if you are supporting CMS in your SE */
#define SE_GDMP_REP_CAT_CMS_PWD   !Top!Secret!
/* Password for Globus Replica Catalog Manager of LHCb. This is COMPULSORY
   ONLY if you are supporting LHCb in your SE */
#define SE_GDMP_REP_CAT_LHCB_PWD  !Top!Secret!
/* Password for Globus Replica Catalog Manager of EarthObservation. This is COMPULSORY
   ONLY if you are supporting EarthObs in your SE */
#define SE_GDMP_REP_CAT_EOBS_PWD  !Top!Secret!
/* Password for Globus Replica Catalog Manager of Biomedical apps. This is COMPULSORY
   ONLY if you are supporting Biomedical in your SE */
#define SE_GDMP_REP_CAT_BIOM_PWD  !Top!Secret!
/* Password for Globus Replica Catalog Manager of WP6. This is COMPULSORY
   ONLY if you are supporting WP6 in your SE */
#define SE_GDMP_REP_CAT_WPSIX_PWD !Top!Secret!


/* WORKER NODE DEFINITIONS ----------------------------------------------
---------------------------------------------------------------------------- */
/* The mountpoint on the WN of the SE exported area via NFS. It should be
   the same used for the SE */
```

```
#define WN_MOUNTPOINT_SE_AREA   CE_MOUNTPOINT_SE_AREA


/* USER INTERFACE DEFINITIONS ------------------------------------------
------------------------------------------------------------------------- */
/* Resource broker */
#define UI_RESBROKER            lngrid06.lip.pt
/* Logging and Bookkeeping URL */
#define UI_LOGBOOK              https://lngrid06.lip.pt:7846
```

Some of the above fields are self-explanatory so we will concentrate on the most important ones. The LCFGSRV gives the name of the LCSFG server and URL_SERVER_CONFIG is the web server, as it was named in a previous paragraph. The Resource Broker host and port are the ones in Lisbon, Portugal. The common site definitions have all the info about the local domain, gateway, site allowed networks, nameservers and netmask. It should be noted that if there are more than one DNS servers, they should appear with a space between them, not a comma, in the nameservers field. The NTP[1] server we chose is one in Sweden. Then we define the SITE_DN, which is the Distinguished Name of your site. For the format of the naming, we can say that it is the standard X.500 naming and the attributes and abbreviations are: Country – C, Locality – L, Organization – O, Organizational Unit – OU, Common Name – CN and Domain Component – DC. For more information, appears elsewhere[2]. Then we define the other hosts of the cluster, but being careful in the naming of many WN. These can be defined explicitly, or by putting stars or brackets, e.g. xg001 or xg00* or xg00[1-5][3]. After that, we have the GIIS info. We are not sure about the TOP_GIIS and TOP_GIIS_HOSTNAME choices; so let the fields as they are.

In the final part of the file, we have the definitions of the Computing Element, Storage Element, Worker Node and User Interface. The corresponding hostnames should be defined correctly; the path and name used for the CE certificate, and security key should be noted. The CE_CERT_SBJ is the information that the RB requests from the Gatekeeper and the format has to be the same as the format of the CA of your country. The RunTimeEnvironment (15) indicates that the site is running but it hasn't been certified yet so one has to choose the value EDG-TEST. Finally, the RB name is the one in Portugal.

The other header file that the `xg001` uses and need checking, is the Linux default resources file `linuxdef-cfg.h`. Usually the only field that has to be changed is the auth.rootpwd, which is the field that creates the root password of the clients. The problem is that the password has to be written in an encrypted manner, for obvious reasons. Since currently the passwords are encrypted with the traditional Unix algorithm crypt, which is based on 3DES encryption, one can use, in a

[1] NTP server – Network Time Protocol servers are servers from which we can get uniform time synchronisation

[2] Deliverable D4.2 – Test and Validation Testbed Architecture

[3] We have not tried this, because we used only one WN

command shell, something like this (in the place of *PassWord*, put your own choice):

```
$ perl -e 'printf crypt("PassWord","Sl")."\n"'
```

The `Sl` is the so-called *salt* of encryption. Now the printed, encrypted password can be inserted into the above field.

The next header file that is included in the list of `xg001`, is the LCFG client specific resources file `client_testbed-cfg.h`. The part of the file, which needs more attention, is:

```
/* Modules configuration */
+update.interfaces          eth0
+update.modlist             3c59x  /* module name of the network card */
+update.mod_3c59x           alias eth0 3c59x
/* Update use /usr/sbin/updaterpms for installing the rpms;
   these are the required parameters. */
+update.rpmcfg              client_testbed
+update.rpmcfgdir           /export/local/linux/6.2/rpmcfg
#define RPMDIR              /export/local/linux/6.2/RPMS
#define EDGDIR              RPMDIR/edg-1.2.2
+update.rpmdir              RPMDIR/release:\
RPMDIR/updates:\
RPMDIR/LCFG:\
EDGDIR/appcommon:\
EDGDIR/appwp10:\
EDGDIR/appwp8:\
EDGDIR/appwp9:\
EDGDIR/ca:\
EDGDIR/edg:\
EDGDIR/external:\
EDGDIR/globus:\
EDGDIR/invalid:\
EDGDIR/lcfg:\
EDGDIR/rh6.2
```

In the update.modlist, the module name of the Ethernet card that is used must be included; in our case, this is 3c59x. The information of this file is extensively used by *update* and *updaterpm* and due to this, the directories where the RPMs are need to be defined. Therefore, the RPMDIR and EDGDIR have to be configured appropriately. Note that in the +update.rpmdir there are no spaces between the directory definitions and no empty text, when changing lines.

What follows, in the structure of the host file xg001, are the header files of users required by VOs (`Users-cfg.h`) and users for CrossGrid (`CGUsers-cfg.h`). These two files remain unchanged. Then it is the main header file, specific for the host and that is `ComputingElement-cfg.h`.

Three points in this file need special attention. These are:

```
globus.gconfline_08          X509_CERT_DIR=/etc/grid-security/certificates

globus.gconfline_11          GRIDMAP=/etc/grid-security/grid-mapfile

+nfs.fs_gridsec                                     /etc/grid-security
```

The first line shows where the globus certificates are stored, the second line indicates that the file grid-mapfile is the one, which is updated with the new authenticated users[1], and the third line shows the export directory for the CA certificates.

In xg001 file exists one more header file, to which we will refer, when we set up the job scheduler (PBS).

The files of the other hosts – xg002 (≡WN), xg003(≡SE) – need no change but the addition of the corresponding main host file header, in explain `WorkerNode-cfg.h` for xg002 and `StorageElement-cfg.h` for xg003.

The file for the UI (xg004) need the creation and addition of en extra header file which during the set up of UI will create additional users besides root. The structure of this file is:

```
/*
  Users-demo-cfg.h
  ======================================================================

*/

EXTRA(auth.groups)                                       xgrid
auth.groupgid_grid                                       2010
EXTRA(auth.users)                             nmastro

+auth.userpwd_nmastro                          nm618EGrZXHv.
+auth.usercomment_nmastro         Nikos Mastroyiannopoulos
+auth.userhome_nmastro            /home/nmastro
+auth.usergroup_nmastro           xgrid
+auth.useruid_nmastro             1501
+auth.usershell_nmastro                        /bin/tcsh
```

Configuration update

All the previous changes we made to the LCFG server have to be propagated to the LCFG profile in XML. The profile files in XML are stored in

`/var/obj/conf/profile/web/profiles/<hostname>.xml`

This file is located on the server and is the file that the client reads during installation or in a case of an update.

The utility we use for creating this XML file is *mkxprof*. If mkxprof is not running as a daemon, we have to issue the following commands (on the LCFG server) to force the creation (or update) of the client XML profiles:

```
$ mkxprof –v –A –R <hostname>
```

[1] We will see in a later paragraph of the guide how this is done

If no hostname is given, then the utility rebuilds the profile configurations of all the hosts. We have noticed that in some cases, when the configuration of a node has changed, we cannot run mkxprof before `touch` the linuxdef-cfg.h file.

Chapter

4

Node Installation and Configuration

As we mentioned in a previous paragraph, before the installation of the nodes of the cluster via LCFG can start, their names and IPs must be registered in the DNS server of the site. This is important because each host looks up for its hostname and IP in the DNS table and if it is not there, the installation cannot continue.

In the next paragraphs, we try to keep the per-element procedure of configuring, for two main reasons. First, we find appropriate to have an installation procedure that will start and finish on the same host, independently of the middleware installed. Besides that, the per-element configuration has the advantage of gathering all the information of one host; also, the various bugs of the installation and configuration are more easily identified.

Computing Element (CE)

The installation starts with the CE (or SE) since it is the host where most of the packages will be installed. Resolving any kind of problems that may occur during this installation, the procedure for the other hosts will be much easier and faster.

Installing system

The ideal case of installing the system in a host will be described first and then we will refer to the problems we faced during this installation

The installation of the system starts with the boot disk of the LCFG, which has been created earlier. After inserting the disk in the drive and while rebooting, one enters the BIOS set-up of the PC. It must be verified that the boot sequence starts with the floppy (usually it is floppy – hard disk – CD) and then exit the BIOS and let the machine reboot from the floppy disk. The first menu is a query about starting nfs. Simple nfs should be chosen and then, the host starts a sequence of steps. It checks its own hostname and IP from a DNS table, it checks the LCFG, web server and nfs and then it starts to get the information it needs about where each package, in the LCFG server, is and if the list it reads is correct or not. What the host reads is the xml profile created earlier in the LCFG server. The CE flags the packages that are about to be installed and then continues with the installation. If everything goes right, after finishing the installation of all the packages, the host will reboot and then you will have a CE ready to work!

However, things don't always go right. In the next few lines, we describe what difficulties we faced and how we overcame them.

The first thing that we noticed is that, if the disk of the host is not "clean", meaning that it is a new disk or a recently formatted one, the fdisk utility, which runs automatically in order to create the needed partitions, does not work properly; it doesn't create the partition table it is supposed to and the system hangs. In a case when the hard disk contains another system, even a Linux one, or a "strange" partition table, is to run fdisk manually, delete every partition on the disk, and renew the partition table.

Even by doing this, the installation on CE did not continue easily. While flagging all the packages for installation correctly, when it came the time for installing globus, we got this message

```
[FAIL] update:rpmRunTransaction failed
       LCFG object update:updaterpms failed [FAILED]
```

and the system stopped. The workaround is quite easy. One has to select first the packages that install Linux and then everything else that remains. The following steps should help as a guide to the process:

A. Change, in CE-rpm, the packages that will be loaded

```
$ cd /opt/local/linux/6.2/rpmcfg
```

Edit the CE-rpm file. Comment the `ComputingElement-rpm.h` and also if you like you can comment all the VO header files, but you don't have to.

B. Touch the Linux definitions configuration file

```
$ cd /var/obj/conf/profile/source
$ touch linuxdef-cfg.h
```

C. Make the profile of the node again

```
$ mkxprof –v –A –R <CE hostname>
```

D. Reboot the node

```
$ sync
$ reboot
```

When installation of Linux on the node completes, we add the ComputingElement-rpm.h in CE-rpm as before (uncomment the line). And we follow the above procedure. In case of having commented a VO header file and you want to install it, after completing the above process, you uncomment it and then in the node (CE) you write:

```
$ /etc/obj/update run
```

The update utility, in a complete and running farm installation, has to be installed in the cron of LCFG. But we will not get into this point right now.

GLOBUS Configuration

The CrossGrid testbed is separated into production, development and test & validation. The first release of the CrossGrid testbed is unified with no separation between the three resources. This is a measure for gaining experience and providing a stable quick interim solution for CrossGrid developers. Thus, the central Grid services for the initial testbed are being provided by LIP. The current set of services is shared by all CrossGrid sites (production, development and test), however in the future most of these services will be made available exclusively to test sites. The following services are currently being hosted at LIP:

Service	Host	Port	Description
Resource Broker (RB)	lngrid06.lip.pt	7771	Central engine for job submission
Logging & Bookkeeping (LB)	lngrid06.lip.pt	7846 15830	Used to maintain information about jobs in the Grid
MyProxy	lngrid07.lip.pt	7512	Provides proxy certificates for long lived jobs
VO server	grid-vo.lip.pt	9990	Hosts the "crossgrid" and "gdmpservers" virtual organizations
Replica Catalogue (RC)	Lngrid08.lip.pt	9980	Hosts information files and location of their replicas

The above services run with EDG version 1.2.2

Resource Broker (RB)

The RB is responsible for the matchmaking between jobs and the existing computing resources provided through gatekeepers. The RB obtains the list of the existing gatekeepers from an LDAP information system located inside the RB system and the list is currently updated manually.

Configuring CE to use the CrossGrid VO

CrossGrid sites should add the crossgrid VO to the local gatekeepers and Storage Elements in order to allow other CrossGrid users to access the site computing resources. This is especially important for the remote users from other countries that are not participating in DataGrid since these users are not included in the DataGrid VOs. Therefore, you must add the crossgrid users to a gatekeeper grid-mapfile. Add the following line to file `/opt/edg/etc/mkgridmap.conf`. The line maps the distinguish names of the CrossGrid user certificates to pooled accounts of the form cgXXX.

```
$ group ldap://grid-vo.lip.pt:9990/ou=testbed1,\
  o=crossgrid,dc=eu-crossgrid,dc=org.cg
```

This entry should be preferably be added after the ones referring to the DataGrid VOs. In this way, CrossGrid users that are registered in DataGrid VOs, they will be mapped to one of these first, otherwise they will be mapped into the CrossGrid VO. If you are using "auth" statements in the configuration file `/opt/edg/etc/mkgridmap.conf`, then you also need to add "auth" line for the CrossGrid VOs. This line is:

```
$ auth ldap://grid-vo.lip.pt:9990/ou=People,\
  o=crossgrid,de=eu-crossgrid,dc=org
```

The next step is to create empty files cgXXX in /etc/grid-security/gridmapdir that they must be owned by root[1].

```
$ cd /etc/grid-security
$ mkdir gridmapdir
$ cd gridmapdir
$ touch cg001
$ touch cg002
$ ...
```

There are some services that have to be started:

```
$ service edg-gridmapfile-upgraded start
$ service edginfo-mds start
$ service edginfo-mds-vo start
$ service globus-mds start
```

If the daemon `edg-gridmapfile-upgrade` is running the **gridmapfile** is automatically rebuilt from the VO servers specified in **mkgridmap.conf**. This means that any modifications made by hand to the gridmapfile will be lost each time the daemon does a rebuild. If the daemon `edg-gridmapfile-upgrade` is not running then any changes performed to the VO servers content or in the **mkgridmap.conf** will not be reflected into the **gridmapfile**.

[1] If this is a security glitch, remains to be shown

Gatekeeper Certificates

For the Globus Gatekeeper service to run, a certificate has to be requested by the designated Certification Authority and installed in the Computing Element. For this to happen one has to generate first a public-private key pair and a certificate request file. The procedure is as follows:

Login to the CE and at the shell prompt enter the following command:

```
$ grid-cert-request –host <CE-FQDN> \
-key /opt/globus/etc/<CEHostname>-computingelement.key \
-cert /opt/globus/etc/<CEHostname>-computingelement.cert \
-req /opt/globus/etc/<CEHostname>-computingelement.req
```

At the above command, replace <CE-FQDN> with the Fully Qualified Domain Name of your own Computing Element (e.g. xg001.inp.demokritos.gr), and <CEHostname> with its hostname (e.g. xg001). Grid-cert-request creates three files in /opt/globus/etc:

- **<CE-Hostname>-computingelement.key**, which contains the private key of the gatekeeper.
- **<CE-Hostname>-computingelement.req**, which contains the public key bundled with additional information, in a so called PKCS#10 format, appropriate for requesting a certificate from a CA.
- **<CEHostname>-computingelement.cert**, which is zero-sized and essentially provides a placeholder for the certificate that will be issued by the CA.

Next simply email the request file (<CE-Hostname>-computingelement.req) to you're the site/VO designated Certification Authority. Probably for this a formal procedure has to be followed, which is defined in the CA Certificate Practice Statement. One has to consult his/her CA responsible for the exact requirements.

The certificate will usually be sent to you, from the CA, by email as an attached file. This file must be saved in /opt/globus/etc with the name <CE-Hostname>-computingelement.cert, overwriting the existing zero-sized file.

Finally, start the gatekeeper:

```
$ service globus-gatekeeper start
```

The Computing Element is now ready to accept remote job submissions.

PBS Configuration

We suppose that the installation of the PBS RPM packages have already installed on CE with LCFG and that PBS services (in /etc/services) have already configured. LCFG can't handle the PBS configuration so it's necessary to manually customize the system.

1. We log in the PBS server (CE) and we add the path to some binary directories of the PBS in the file /usr/spool/PBS/pbs_environment

```
PATH=/bin:/usr/bin:/usr/pbs/bin:/usr/pbs/sbin
LANG=en_US
```

2. We check if the PBS daemon is running, if it is running, we stop it

```
$ service --status-all
$ /etc/rc.d/init.d/pbs stop
```

3. We create and edit the file /usr/spool/PBS/server_priv/nodes adding for every Worker Node a line stating the total number of Virtual Processors like this:

```
<WorkerNodeFQDN> np=2
```

4. We edit[1] the configuration file /usr/spool/PBS/pbs_server.conf that will be given as input to the **qmgr** utility

```
# Create queues and set their attributes
#
# Create and define queue work
create queue workq
set queue workq queue_type = Execution
set queue workq max_running=12  #total cpu number
set queue workq enabled = True
set queue workq started = True
#
# Set server attributes
set server scheduling = True
set server managers = admin@host
set server managers +=root@<computing-element-name>
set server default_queue = workq
set server log_events = 511
set server mail_from = adm
set server query_other_jobs = True
set server resources_default.neednodes = 1
set server scheduler_iteration  = 600
set server node_pack = False      # important for job distribution
```

5. Make a backup of the nodes file

[1] Some instructions and a test file can be found http://gridportal.fzk.de/cgi-bin/viewcvs.cgi

```
$ cp /usr/spool/PBS/server_priv/nodes /usr/spool/PBS
```

6. We "feed" the qmgr with the above configuration file

```
$ /usr/pbs/sbin/pbs_server -t create
$ /usr/pbs/bin/qmgr < /usr/spool/PBS/pbs_server.conf
```

Important Notice: Check the file /usr/spool/PBS/server_name. This file should have the hostname of your PBS server, meaning the CE hostname. In any other case, the above input of the qmgr program will give the following message

```
    $ Unknown host
    $ qmgr: cannot connect to server
```

7. Then you can restart the pbs daemon (restore also the `nodes` file because it keeps deleting after the feed!!):

```
$ cp /usr/spool/PBS/nodes /usr/spool/PBS/server_priv
$ /etc/rc.d/init.d/pbs start
```

Storage Element (SE)

Installing System

The procedure of installing the system in the Storage Element is exactly the same as the one followed to the CE installation. Therefore, we write only a bulleted list either for a case things work out as expected or the case where you face the same problems as in the CE case.

- Insert boot disk in SE
- Run NFS – The system checks the network, IPs, hostname, flags the packages to be installed and in the end starts installing
- SE reboots and then installation completes

This is the optimistic case! If things don't go that good, follow the instructions.

1. Change, in `/opt/local/linux/6.2/rpmcfg/`**`SE-rpm`**, the packages that will be loaded. Comment the `StorageElement-rpm.h`, so that it won't be loaded the first time you try the installation.

2. Touch the Linux definitions configurations file

   ```
   $ cd /var/obj/conf/profile/source
   $ touch linuxdef-cfg.h
   ```

3. Make the profile of the node (xml file) again

```
$ mkxprof -v -A -R <SE hostname>
```

4. Reboot the node

```
$ sync
$ reboot
```

5. After this step, Linux is installed on your system. Add the StorageElement-rpm.h in SE-rpm (uncomment the line) and do as above.

Globus Configuration

As it has been explained in the Globus configuration of CE, the following steps have to be followed so as to add CrossGrid users to your site's Storage Element servers.

1. To add crossgrid users to a SE grid-mapfile you need to add the following line to the configuration file `/opt/edg/etc/mkgridmap.conf`

```
$ group ldap://grid-vo.lip.pt:9990/ou=testbed1,\
  o=crossgrid,dc=eu-crossgrid,dc=org.cg
```

2. As all Storage Elements run GDMP, you need to add a line to map the host certificates, something that is required by GDMP. The distinguish name of all CrossGrid SE running GDMP needs to be mapped to the account gdmp in each SE. The line is:

```
$ group ldap://grid-vo.lip.pt:9990/ou=apptb,\
  o=gdmservers,dc=eu-crossgrid,dc=org gdmp
```

   It is preferable, these lines to be added after the ones referring to the DataGrid VOs.

3. If you are using "auth" statements in the configuration file /opt/edg/etc/mkgridmap.conf then you also need to add "auth" lines for the crossgrid VOs.

```
$ auth ldap://grid-vo.lip.pt:9990/ou=People,\
  o=gdmservers,dc=eu-crossgrid,dc=org
```

4. Create empty files cgXXX, corresponding to users – 20 is usually enough number – in /etc/grid-security/gridmapdir

```
$ cd /etc/grid-security
$ mkdir gridmapdir
$ cd gridmapdir
$ touch cg001
$ touch cg002
$ ...
```

5. Start services

```
$ service edg-gridmapfile-upgraded start
$ service edginfo-mds start
$ service edginfo-mds-vo start
$ service globus-mds start
```

SE Gatekeeper Certificates

As with the Computing Element and since the Storage Element E hosts a Globus Gatekeeper, one has to request and install a digital certificate for the service. The procedure is similar to the one for the CE:

Login to the SE and at the shell prompt enter the following command:

```
$ grid-cert-request –host <SE-FQDN> \
-key /opt/globus/etc/<SEHostname>-storageelement.key \
-cert /opt/globus/etc/<SEHostname>-storageelement.cert \
-req /opt/globus/etc/<SEHostname>-storageelement.req
```

At the above command replace <SE-FQDN> with the Fully Qualified Domain Name of the Storage Element (e.g. xg004.inp.demokritos.gr), and <SEHostname> with its hostname (e.g. xg004). Grid-cert-request will create three files in /opt/globus/etc:

- **<SE-Hostname>-storageelement.key**, which contains the private key of the gatekeeper.
- **<SE-Hostname>-storageelement.req**, which contains the public key bundled with additional information, in a so called PKCS#10 format, appropriate for requesting a certificate from a CA.
- **<SE-Hostname>-storageelement.cert**, which is zero-sized and essentially provides a placeholder for the certificate that will be issued by the CA.

Then email the request file (<SE-Hostname>-computingelement.req) to the site/VO designated Certification Authority. When the certificate has been received from the CA it has to be copied in /opt/globus/etc with the name <SE-Hostname>-storageelement.cert, overwriting the existing zero-sized file.

Finally start the gatekeeper:

```
$ service globus-gatekeeper start
```

Worker Node (WN)

Installing System

The WN installation follows the path of both CE and SE. We will restrain ourselves to referring only the basic steps

1. Change, in `/opt/local/linux/6.2/rpmcfg/`**`WN-rpm`**, the packages that will be loaded. Comment the `WorkerNode-rpm.h`, so that it won't be loaded the first time you try the installation.

2. Touch the Linux definitions configurations file

```
$ cd /var/obj/conf/profile/source
$ touch linuxdef-cfg.h
```

3. Make the profile of the node (xml file) again

```
$ mkxprof -v -A -R <WN hostname>
```

4. Reboot the node

```
$ sync
$ reboot
```

5. After this step, Linux is installed on your system. Add the WorkerNode-rpm.h in WN-rpm (uncomment the line) and do as above.

PBS Configuration

On each WN a job submission system has to be installed and configured properly. We still don't have an LCFG object to do this automatically, but following the latest testbed setup agreed inside the integration team (on exporting the home dir all over the WN), there is little to be done.

1. In WN, as in the PBS server (typically CE), you have to add the path to some binary directories of the PBS in the file /usr/spool/PBS/pbs_environment

```
PATH=/bin:/usr/bin:/usr/pbs/bin:/usr/pbs/sbin
LANG=en_US
```

2. We check if the PBS daemon is running, if it is running, we stop it

```
$ service --status-all
$ /etc/rc.d/init.d/pbs stop
```

3. Then you have to specify the name of the queue server (typically the hostname of CE) into the file

```
/usr/spool/PBS/serve_name
```

4. Finally edit the file /usr/spool/PBS/mom_priv/config this way:

```
$clienthost <ComputingElementFQDN>
```

```
$logevent 0xlff
$usecp <ComputingElementFQDN>:/home /home
```

5. Then you can restart the pbs daemon:

```
$ /etc/rc.d/init.d/pbs start
```

User Interface (UI)

Installing System

The User Interface was the node that we had the least problems. During the installation, everything went as planned and there was no reason to change the set-up file, UI-rpm in LCFG. This is because the UI install a few packages that cannot be compared to the ones of CE or SE. In case you face any problems follow the below mentioned instructions:

1. Change, in `/opt/local/linux/6.2/rpmcfg/`**`UI-rpm`**, the packages that will be loaded. Comment the `UserInterface-rpm.h`, so that it won't be loaded the first time you try the installation.

2. Touch the Linux definitions configurations file

```
$ cd /var/obj/conf/profile/source
$ touch linuxdef-cfg.h
```

3. Make the profile of the node (xml file) again

```
$ mkxprof -v -A -R <UI hostname>
```

4. Reboot the node

```
$ sync
$ reboot
```

5. After this step, Linux is installed on your system. Add the UserInterface-rpm.h in UI-rpm (uncomment the line) and do as in the case of CE or SE.

Post-install Configuration

The User Interface is the machine that one uses to submit jobs to the Grid. The jobs submitted pass through a Resource Broker and then to an available and appropriate Worker Node somewhere on the Grid. Up to now, the RB used is the one in Portugal (LIP), so to configure a UI to submit jobs through the RB located at LIP, some changes must be made to the UI configuration file located at `/opt/edg/etc/UI_ConfigENV.cfg` in each UI system.

A. Change the LB to point to the system **lngrid06.lip.pt**

```
%%beginLB%%
https://lngrid06.lip.pt:7846
%%endLB%%
```

B. Change the RB to point to the system **lngrid06.lip.pt**

```
%%beginRB%%
https://lngrid06.lip.pt:7771
%%endRB%
```

The User Interface has to be configured to use the Replica Catalogue located at LIP, so a few changes have to be made to the RC configuration file located at `/opt/edg/etc/rc.conf` in each UI system.

```
RC_REP_CAT_MANAGER_DN=cn=Manager,dc=lngrig08,dc=lip,dc=pt
RC_REP_CAT_MANAGER_PWD=Testbed1RC
RC_REP_CAT_URL=ldap://lngrid08.lip.pt:9980/rc=\
CrossGridReplicaCatalogue,dc=lngrid08,dc=lip,dc=pt
RC_LOGICAL_COLLECTION=ldap://lngrid08.lip.pt:9980/lc=cgtst0,\
rc=CrossGridReplicaCatalogue,dc=lngrid08,dc=lip,dc=pt
```

The above example is for CrossGrid VO only. For other VOs the parameter lc=cgtst0, which is in bold, must be replaces accordingly.

VO	**LC**
CrossGrid	cgtst0
WP6	wpsixtst0
Atlas	atlaststst0
CMS	cmststst0

Chapter

5

Maintenance and Upgrading

The end of the Installation and Configuration procedure marks the beginning of the next long-term cycle, which is the daily base maintenance and upgrading of a Grid cluster. Here LCFG demonstrates its power as a cluster maintenance tool. In general the task of maintaining a cluster up-to-date and trouble free, proves to be much easier than the installation of the cluster per se. Most of the tasks are carried out centralized from the LCFG server and there are only a few occasions that one has to actually login to a specific host to perform an action.

Installing New Software

There are situations when one has to install or upgrade an individual package at a specific host or at a group of hosts. For instance when a new Certification Authority joins the Testbed the CA certificate, CRL and signing policy files must be installed on the CE and SE hosts, so that users with certificates signed from this CA can exploit the cluster's resources.

To install a new package you'll have to download and copy the new package into the appropriate LCFG server directory of your rpm installation tree (e.g. /opt/local/linux/6.2/RPMS/edg-1.2.2/ca). Then run *make* in this directory to create the rpm headers for the newly copied package. This is important because otherwise LCFG update tools won't be able to recognize the existence of the new package. Next you'll have to edit the appropriate file in /opt/local/linux/6.2/rpmcfg and add the name of the package to the list. For the **ca** example, you will have to edit security-rpm.h. That's all there is to it.

To force an update of a node, login to that specific node and issue the command:

```
$ /etc/obj/updaterpms run
```

updaterpms will check the rpm list from the LCFG server and install all new packages listed.

For the upgrade of an individual package the procedure is almost the same. Copy the rpm into the appropriate directory, delete the old version of the package, run make, and edit the suitable rpmcfg file. This time don't add a new entry but change the existing entry from the old version name to the new one (e.g. pckg-v1.0 to pckg-v1.1). Again to force an upgrade, run the *upgraderpms* script in the specific node. This time *upgraderpms* will inform that the package will be upgraded and not installed since it already exists in the local rpms database.

Upgrading to new EDG versions

From time to time, every couple of weeks nowadays, the good guys in the EDG development team release a new version of DataGrid's software collection. The process of upgrading the cluster to the new version is similar with the process of installing it at the first time, only simpler since now one has only application software that needs to be installed/upgraded (this is the case at least until EDG migrates to RH7.2). As mentioned in Chapter 3, we suggest keeping each EDG version in separate directories. This will consume some additional disk space but keeping things clean and simple is much more important in an administration process.

For example if a new version (say edg-1.2.3) is released login to the LCFG server and issue:

```
$ cd /opt/local/linux/6.2/RPMS
$ mkdir edg-1_2_3
$ cd edg-1_2_3
```

Next, check the DataGrid's RPM repository and create as many directories as are the package categories appearing there (globus, ca, external etc). Copy the Makefile, as you did when you were installing the cluster the first time, from say the /opt/local/linux/6.2/RPMS/release, in each one of the newly created directories. Next `cd` to each directory and `wget` the corresponding file using the command as it is shown in the repository web page. For instance, in the globus directory enter:

```
$ wget -r -nd \
http://datagrid.in2p3.fr/autobuild/rh6.2/rpmlist/CE-globus-test_v1_2_3.html
```

This will download the globus related packages for CE. Go on and download the entire packages for the node components supported in your site (CE, SE, UI etc) as it was described in Chap 3. When finished, don't forget to run *make* in each directory to generate the header rpm files for the new packages.

Now that all the packages are in the repository go to the source directory at /opt/obj/conf/profile/sources and edit client_testbed-cfg.h. Change the EDGDIR macro to point to the directory where the new version has been downloaded. e.g. :

```
#define EDGDIR          RPMDIR/edg-1.2.3
```

In addition, check the directory structure listed after the above directive, which is defined in the update.rpmdir variable, to ensure that it matches the one you have created in your repository.

Now at the shell prompt in the source directory issue:

```
$ touch linuxdef-cfg.h
$ mkxprof –v –A node*
```

mkxprof should be run against all node profiles that are being upgraded (usually all of them). This will initiate an automatic upgrade of all nodes. If one wants to specifically force an upgrade in a specific host, and have the chance to monitor it, he/she has to login to this host as root and call the upgraderpms LCFG client script as follows:

```
$ /etc/obj/updaterpms run
```

User Accounts

There are two situations when new user accounts have to be created in the cluster. The first case is when one has to add accounts for a new Grid VO and the second, when a new user is registered in one's site. In both cases, the necessary administration actions are performed centrally in the LCFG server. In either case, one can still login to a specific host and create an account with a command like *adduser* but this account will go away with the next reboot of the node, so it should be avoided.

Grid VO Accounts

When a site joins a new Virtual Organization (VO), usually a number of accounts must be created on all CE, WN and SE nodes. The new accounts are declared in the appropriate header file in /var/obj/conf/profile/source. If you have followed the instructions in chapter 3, the file you have to edit is Users-cfg.h. First, the new accounts require a group to be created:

```
EXTRA(auth.groups)              exmpl
auth.groupgid_exmpl             2002
```

The second line assigns a group ID to the newly created group. Next, declare the names of all new accounts:

```
EXTRA(auth.users)  exmpl001 exmpl002 exmpl003 exmpl004 exmpl005
```

Then, for each new account provide additional information:

```
+auth.usercomment_ exmpl001          put a comment here
+auth.userhome_exmpl001              /home/exmpl001
+auth.usergroup_exmpl001             exmpl
+auth.usersuppgroups_ exmpl001       othergrp
```

```
+auth.useruid_ exmpl001                2001

+auth.usercomment_ exmpl002            put a comment here
+auth.userhome_exmpl002                /home/exmpl002
+auth.usergroup_exmpl002               exmpl
+auth.usersuppgroups_ exmpl002         othergrp
+auth.useruid_ exmpl002                2002
```

and so on…

The first line (**auth.usercomment**) usually contains the name of the user, but since these are polling accounts, that are going to be used by various users, you can put there whatever comment you want. **auth.userhome** points to the home directory that will be created in the Computing Element. **auth.usergroup** is of course the primary group that this user belongs to. You can also put additional groups using the **auth.usersuppgroups** directive. Finally the **auth.useruid** is the **unique** id for the user. For user naming you should pick a uniform naming scheme like groupXXX, where group is the group name the user belongs to and XXX an incrementing number.

As we mentioned, these accounts will be created in each CE, WN and SE node of the cluster. The home directory of each account is created only in the CE and then shared with NFS to all other nodes. To do that, edit the UsersNoHome-cfg.h file and add the line:

```
auth.usernotcreatehome_exmpl001 true
```

for each new user. This prevents the home directory from being created in hosts other than the CE.

Now you are ready to update the node profiles. In the source directory run:

```
$ touch linuxdef-cfg.h
$ mkxprof –v –A node*
```

and that's it; in a couple of minutes, all new accounts will be created automatically.

For the last step login to the CE and touch the corresponding files in *gridmapdir* for each new account, otherwise job submission to the Gatekeeper for these accounts will fail. So login to the CE and:

```
$ cd /etc/grid-security/gridmapdir
$ touch exmpl001 exmpl002 ...
```

User Interface Accounts

Creating and account in the UI node is similar with VO accounts, only simpler this time. The new user is declared only for the UI, giving the details (group, id, user name) as in the previous paragraph. Usually there should be a UIUsers-cfg.h file dedicated to UI user account definitions and included only in the UI profile.

The difference with VO accounts is that, this time, additional information is provided: a password and the initial login shell. The password is given in an encrypted form, produced using the

```
$ perl –e 'printf crypt("PassWord","Sl")."\n"'
```

command as when defining the root password in linux-def.cfg. For instance to create a user named john the entry in UIUsers-cfg.h should look like:

```
+auth.userpwd_john              c3fx2l1I0qISY
+auth.usercomment_ john        John Smith
+auth.userhome_ john           /home/john
+auth.usergroup_ john          users
+auth.useruid_ john            1501
+auth.usershell_ john          /bin/tcsh
```

At auth.usercomment provide the full name of the user. This is important and needed when the user will later on try to request a grid certificate from his account. The auth.usergroup field should contain an existing usergroup in the UI node.

After appending the new account info in this file, update the UI profile:

```
$ touch linuxdef-cfg.h
$ mkxprof –v –A uiprofile
```

and the new user will be created in UI, at the next UI profile pulling (usually 2-3 minutes).

References

1) "The Anatomy of the GRID: Enabling Scalable Virtual Organizations" Ian Foster, Carl Kesselman, Steven Tuecke (to appear: Intl. J. Supercomputer Applications, 2001)

2) "Large scale Linux Configurations with LCFG" Paul Anderson, Alastair Scobie

3) "LCFG: The Next Generation" Paul Anderson, Alastair Scobie

4) "Getting started with LCFG" Paul Anderson Jessie Paterson DICE Computing Environment Project

5) "LCFG lite: a reduced kit for Grid middleware" Enrico Ferro http://datagrid.in2p3.fr/distribution/datagrid/wp4/installation/dec/lcfg-lite/lcfg-lite.html

6) "Datagrid testbed: notes about farm installation by means of LCFG" Massimo Biasotto, Andrea Chierici, Enrico Ferro, Marco Serra http://www.lnl.infn.it/datagrid/wp4-install/testbed-lcfg/testbed-lcfg.html

7) "WP4 LCFG FAQ" Massimo Biasotto, Andrea Chierici, Enrico Ferro, Marco Serra

8) "LCFG Site Server Installation Guide" Mohammad Jaudet, Julian Blake, Jan Iven

9) "First Time LCFG Server Installation: red Hat 6.2 with Updates" Julian Blake

10)"CrossGRID services at LIP" http://www.lip.pt/computing/projects/crossgrid/crossgrid-services/

11)"The CrossGrid Test and Validation testbed" http://www.lip.pt/computing/projects/crossgrid/crossgrid-tv/services.htm